ARTIST Methodology and Framework: A novel approach for the migration of legacy software on the Cloud

Andreas Menychtas, Christina Santzaridou, George Kousiouris, Theodora Varvarigou
National Technical University of Athens
Athens, Greece
{ameny, csantz, gkousiou}@mail.ntua.gr, dora@telecom.ntua.gr

Leire Orue-Echevarria, Juncal Alonso
ICT - European Software Institute Division
TECNALIA
Zamudio, Spain
{Leire.Orue-Echevarria, Juncal.Alonso}@tecnalia.com

Jesus Gorronogoitia
Service Engineering & IT Platforms Lab
ATOS Research & Innovation
Madrid, Spain
jesus.gorronogoitia@atosresearch.eu

Hugo Bruneliere
AtlanMod Team
Inria, Mines Nantes & LINA
Nantes, France
hugo.bruneliere@inria.fr

Oliver Strauss, Tatiana Senkova
Software Technology Team
Fraunhofer Institute for Industrial Engineering IAO
Stuttgart, Germany
{oliver.strauss, tatiana.senkova}@iao.fraunhofer.de

Bram Pellens, Peter Stuer
Spikes N.V.
Brussels, Belgium
{bram.pellens, peter.stuer}@spikes.be

***Abstract*—Nowadays Cloud Computing is considered as the ideal environment for engineering, hosting and provisioning applications. A continuously increasing set of cloud-based solutions is available to application owners and developers to tailor their applications exploiting the advanced features of this paradigm for elasticity, high availability and performance. Even though these offerings provide many benefits to new applications, they often incorporate constrains to the modernization and migration of legacy applications by obliging the use of specific development technologies and explicit architectural design approaches. The modernization and adaptation of legacy applications to cloud environments is a great challenge for all involved stakeholders, not only from the technical perspective, but also in business level with the need to adapt the business processes and models of the modernized application that will be offered from now on, as a service. In this paper we present a novel model-driven approach for the migration of legacy applications in modern cloud environments which covers all aspects and phases of the migration process, as well as an integrated framework that supports all migration process.**

Keywords—Cloud Computing, Legacy Software, Modelling, Migration, Modernization, Methodology

I. INTRODUCTION

Clouds [1] have changed the way that computing, storage and networking resources are purchased and consumed by providing advanced services across the cloud stack layers [2]. Those are capable to support the needs of several application types from big enterprises and SMEs, to the public sector and academia. Clouds are not only solutions for developing applications, but have also leveraged new business models and value chains, that actually transform them into true business ecosystems. The variety of cloud offerings on both technical and business level is simplifying the development process, reducing the time to market and allowing a better control on the capital and operational expenditures of the applications offered as services.

However, legacy applications have some unique characteristics which introduce many challenges and difficulties in processes of their modernization and migration to cloud environments. This is two-folded; on one hand, there are technical issues related with the nature of the specific application and on the other, the business aspects that need to be considered in offering an application "as a service". Often the legacy applications are not cloud-enabled, following monolithic architecture design approaches and implemented in technologies which may be deprecated and cannot be supported in a "plug 'n play" manner. The modernized version of the applications necessitates the equivalence of functionality and performance as well as business continuity. This does not only require adaptation of the software and integration of modern services of cloud solutions on a technical level (monitoring, security etc.), but also changing the business processes and models based on which the application is offered to the customers so as to exploit the strategic advantages of clouds.

In this process several questions may arise; what is the required effort and cost for the migration? Is the migration

possible for all of the application features? What architectural design should the modernized application have? Which are the pros and cons of each option? What is the best target environment for my application? And most importantly, what are the exact steps that should be followed and how?

In this work, we present a model-driven [3] modernization and migration approach developed in the frame of ARTIST EU Project [4] which is not only capable to answer the aforementioned questions for most legacy applications, but also provides an innovative toolbox that supports developers in each migration task. ARTIST will deliver a "one stop shop" for the migration of legacy applications that tailors a generic methodology to the requirements of the particular application and instantiates it, guiding all involved actors through the various tasks in an integrated environment that realizes them.

ARTIST Methodology and Framework considers both the technical and business aspects of the legacy applications and covers not only the core migration part of the process involving technical, business model and organizational process issues but also the pre-migration, where based on the assessment of the initial and target situations of the application added to the results of a technical and business feasibility analysis, the various steps and tasks of the methodology are customized, as well as the post-migration where the outcomes are validated and certified. In addition, ARTIST allows for migration artefacts to be reused and evolve in order to improve the migration effectiveness for the specific or future legacy applications in terms of cost, effort and performance.

In section II we analyse similar approaches while in section III we explain the motivation for our work. The ARTIST Migration Methodology is described in detail in section IV and the architecture of the ARTIST framework in V. Finally in section VI the conclusions of our work are presented.

II. Related Work

Prior to the definition of the various ARTIST Migration Methodology elements, the ARTIST team has examined the related migration approaches and extracted some interesting findings, being the most important one that there is no methodology covering all migration processes/phases required in the ARTIST approach.

SMART [6] approach is relevant in the pre-migration phase while trying to understand the system operation in order to be able to identify and analyse the gap between the legacy system and the desired one. However, most of this analysis is performed manually and based on the knowledge of the participating team. After acquiring that knowledge, SMART proposes an ad-hoc migration strategy created for each system. In addition, since SMART is focused on the migration to SOA, many relevant issues that concern the basics of SaaS architectures are not treated.

XIRUP methodology [7] was proposed in MOMOCS Project. XIRUP is considered as a general-purpose MDE-based modernization methodology, not specifically designed to address the challenges of migrating legacy applications to cloud environments. In particular, XIRUP is a feature-driven modernization methodology, where the whole legacy system is decomposed into features (as offered by encapsulated components) that are iteratively evaluated (for migration decision support), migrated and assessed (post-migration evaluation). This approach could fit well when a partial migration is required (e.g. hybrid Cloud environment, including some migrated components coexisting with some legacy components). XIRUP also defines method fragments as well isolated engineering processes for software modernization, in line with the Situational Methods Engineering techniques [8] and assigns groups of methods fragments to concrete XIRUP methodology phases. Nonetheless, even if some of these methods fragments are applicable to foreseen ARTIST methodology phases, they need to be aligned and extended to cope with the particularities of the migration to the Cloud.

OMG ADM (Architecture-Driven Modernization) [9] provides a set of generic metamodel specifications that could be relevant within the context of ARTIST. The Software Measurement Metamodel (SMM) proposed by this methodology, could be used while dealing with assessment (e.g. for expressing appropriate metrics/measures as well as representing the result of their computation), the Knowledge Discovery Metamodel (KDM) and to a lower extent the Abstract Syntax Tree Metamodel (ASTM) can be used for reverse engineering purposes (e.g. for representing the legacy source code as models in a "neutral" technology-independent manner). The proposed two-steps Model Discovery + Model Understanding approach and corresponding MoDisco tooling [12] is simply covering the reverse engineering phase. However, it can be reused and extended accordingly within the context of the more global ARTIST approach.

REMICS methodology [10], while following an MDE approach, does not take into consideration non-functional requirements (i.e. performance) inherent to SaaS applications and neither addresses architectural issues such as multi-tenancy or scalability. REMICS relays the monitoring, billing and security issues to the Cloud provider where the application is deployed on. The business issues (business model and processes), related to the components mentioned before (billing, monitoring) are also not considered. In addition, REMICS executes the migration on brute force, without considering the feasibility or convenience to migrate.

mCloud approach [11] comes with an analysis of the application maturity in terms of business model and technology as well as a cost benefit analysis. However, the core migration part of the methodology is not following MDE techniques.

III. Motivation

Unlike current methodologies, ARTIST presents an end-to-end approach. As it will be explained later in detail, ARTIST starts with a feasibility analysis compared to existing approaches where this either comes too late, i.e.; when the decision to migrate has been taken, or is generally ignored. Ignoring such issues mean repeating errors which can imply extremely high expenditures, projects that never end and even worse, some (very expensive) failures. To stop a migration project on time is often cheaper, from both a business and managerial perspective. This is especially critical in the current economic crisis where money resources are more limited within companies.

Additionally, existing modernization methodologies mainly focus on the technical aspects. However, moving to the Cloud implies a business model shift. Thus, business model decisions can be closely related to more technical decisions. Changing the way the company makes business not only impacts the daily business of the company in terms of processes but also affects the application itself, e.g. business decisions such as pricing per use or per number of users mean that the application has to be monitored at all times and the bills have to be created in an automated way. Existing methodologies to migrate to the Cloud overlook this issue, since they are more focused on the architectural challenges.

Once an application has been migrated, it needs to be provisioned. Provisioning an application as a service is not the same as delivering it as a product. While the latter involves installation or consultancy services, the former involves for instance, more customer service, regular updates or new roles that come into play. The studied methodologies do not directly address this issue, while the ARTIST Methodology intends to consider it a fundamental point.

Thus, the added value of the ARTIST Methodology can be summarized as follows: 1) it includes a feasibility analysis before any investment is actually made, 2) it is focused on Cloud-compliant architectural issues at both application and infrastructure levels, 3) it includes business model issues that are strongly linked to the technical decisions that are made, 4) it takes into account the impact of the business model shift on the organization processes, 5) it fosters reusability and automation, 6) it globally prepares the software for its evolution.

IV. ARTIST Migration Methodology

A. Methodology Overview

The ARTIST Methodology consists of three (plus one) major phases which are analysed below. These phases are:

Pre-migration: In this phase a study of the technical and economic feasibility will be conducted as a prerequisite to the migration/modernization of the legacy system.

Migration: This phase will perform the migration process itself, by using both reverse engineering (RE) and forward engineering (FE) techniques in order to deploy the legacy system in the Cloud. It should be noted that based on the particular legacy application requirements and the *pre-migration* analysis results, the migration processes and their flow are explicitly customized. The business model concerns are also studied and defined in this phase. These business issues are included in the application architecture (e.g. through monitoring and billing components). Organizational processes to face the new situation are also (re)defined in this phase. Finally, *migration phase* includes the verification and validation (V&V) of the final system.

Post-migration: In this phase, the modernized application components will be deployed onto the target environment and it will be checked if both the technical and business objectives established in the pre-migration phase have been achieved. The validation activities that are foreseen are mainly focused on behavioural equivalence, model based testing and end-user functional and non-functional testing. Moreover, a certification model will be created in order to increase customer confidence in the SaaS system.

Migration Artefacts Reuse & Evolution: This phase includes all needed application maintenance activities after migration to the Cloud, such as software updates Cloud provider changes, etc.

The following picture summarizes the different phases of the ARTIST Migration Approach.

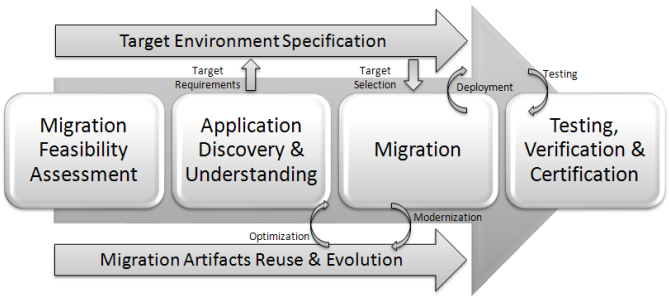

Fig. 1. ARTIST Methodology Overview

B. Development of the Methodology

One of the main tasks within the project has been the definition of the roadmap elements (methodology, steps, timing, guidelines, etc.) that the ARTIST users should follow to perform the migration of their legacy products. The key outcome is a detailed, but also generic, methodology that covers all migration tasks and processes.

ARTIST is defining a generic methodology that includes all tasks. Based on the obtained in the first phase of the methodology results on legacy application migration feasibility analysis, the pre-migration methodology for the migration and provisioning will be instantiated and customized for that specific project (Fig. 2).

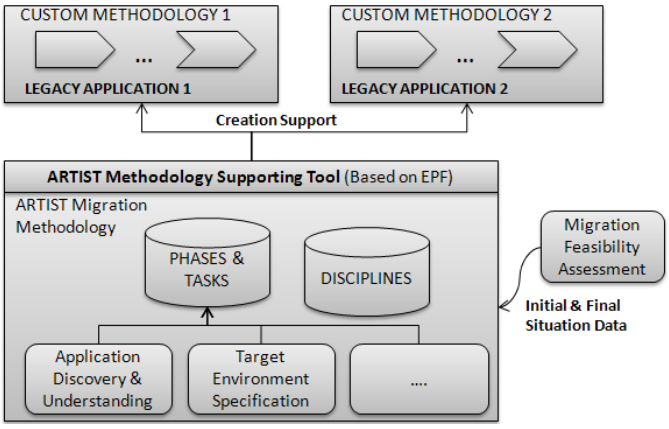

Fig. 2. ARTIST Methodology Instantiation

For the effective definition of the methodology the following aspects were identified:

- **Phase:** Phases are the highest abstraction level of the different steps composing the methodology.

- **Task:** Each phase is composed of several tasks. These tasks can be related to the technical activities performed in the course of the project related to Infrastructure modelling and performance analysis, Reverse Engineering, Forward Engineering and the recommendations at business-level given in the pre-migration phase.
- **Disciplines:** Each phase and task will be identified as technical, process or business. Disciplines in this case are like categories:
    - **Technical tasks** are those related to the technical activities of the migration, considering the legacy code / models and the target platform.
    - **Business tasks** are those related to the activities to be performed in order to update the business model to the new product or service.
    - **Process tasks** are those related to the activities required to accommodate the business processes of the company to the new situation.

The overall ARTIST Methodology is formally described using the Eclipse Process Framework – EPF [15] and SPEM2.0 [16] specification and the generated diagrams of each phase are presented in the following sections.

C. Pre-Migration Phase

As stated in [5], the pre-migration phase (Fig. 3) is the starting point of each migration. Migration of legacy applications is considered as a very challenging project that involves not only changing the way companies are going to deliver their software, but also their business model, and how the company is organized in terms of processes. Thus, software vendors need to analyse if what they want to achieve, is actual feasible to them both technologically and economically.

The first step in this pre-migration phase is to analyse how **mature** the application is in terms of technology (i.e. architecture, programming language, database, integration with 3rd party offerings, installation requirements, versioning, etc.) and business (i.e. current business model, existence of SLA, maintenance and upgrades procedures, customer service, etc.). It also considers how the customer wants the application to be in those two axes (i.e. architectural design, cloud provider requirements, business model, legal concerns, performance thresholds values, etc.) once the application is migrated. The analysis of both the current and ideal situations allows ARTIST to perform a gap analysis, described in terms of a technical feasibility analysis and a business feasibility analysis.

The **technical feasibility analysis** is aimed to provide a snapshot of the application's design quality, of its complexity and coupling, etc. This is realized thanks to reverse engineering techniques and a static analysis of the source code. These data, in combination with the ideal technological maturity identified in the maturity assessment as well as with the target platform requirements, will provide some metrics such as how much effort will be needed to perform this migration and so on.

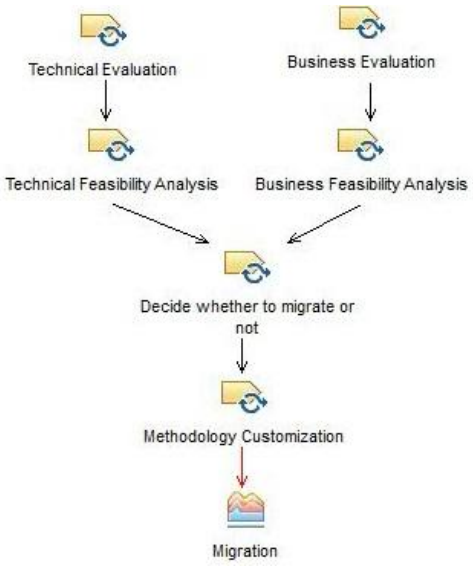

Fig. 3. Pre-migration EPF Model

Furthermore, the feasibility analysis, in combination with the results from the ideal situation identified in the maturity assessment and the identification of the target platform expected characteristics, a **business feasibility analysis** is performed. This business feasibility analysis is aiming to provide not only economic information (ROI, payback, etc.) but also what are the main risks to be faced with the migration and the organizational processes affected by the uptake of the new business model. The results obtained in both the feasibility and business analysis will guide decision makers to identify the most appropriate strategy in terms of migration. In this stage information about target platforms will also be used in order to determine aspects such as the cost of the deployed application, which may vary depending on the selection of the target services/platforms to use.

D. Migration Phase

The Migration phase is the core part of the ARTIST Methodology, incorporating unique features for reverse and forwards engineering as well as for the effective target environment specification.

1) Application Discovery & Understanding

During the technical feasibility assessment as well as during the migration process itself, the discovery of models describing the legacy system is first required in order to have a better understanding of it. Thus, one of the main goals of Model Driven Reverse Engineering (MDRE) is to extract the overall logic of the legacy system (and some implementation details when necessary for the actual migration), considering different abstraction levels depending on the targeted stakeholders. In any case, such a MDRE process is composed of the two main tasks (Fig. 4):

Model Discovery generates the minimum set of required "raw" initial models out of (some of) the legacy artefacts composing the system. This first task notably implies analysing

the different available legacy artefacts to identify the ones which are actually relevant to the considered migration scenario. This preliminary action can be performed with the help of the *Taxonomy of Legacy Artefacts* as established and developed within the timeframe of the ARTIST Project.

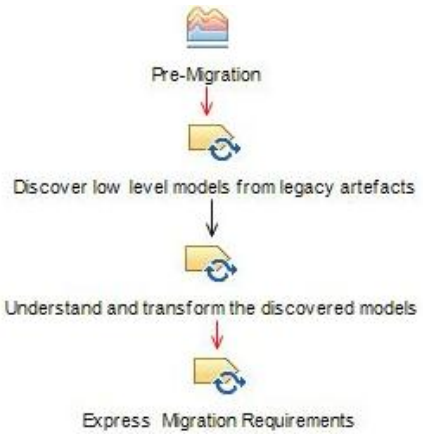

Fig. 4. Application Discovery and Understanding EPF Model

Model Understanding produces the necessary "processed" derived models, thus filtering only the information required for the remainder of the overall process (i.e.; Modernization notably). This second task relies on the use of various (chains of) model-to-model transformations from the previously obtained "raw" initial models to the final derived models to be provided as inputs of the Modernization tasks. These derived models may conform to different metamodels and so represent "views" on the legacy system at different levels of abstraction, according to the requirements of the next tasks.

2) Target Environment Specification

In the Target Environment Specification phase (Fig. 5) we have two parallel processes taking place, one in the application domain and one in the candidate target environment domains. Following the analysis of the various application features and the creation of models describing the legacy system using reverse engineering techniques, the performance aspects of each application element and feature are examined and profiled. In this process, these aspects are linked with specific software solutions exploiting trace analysis and benchmarking, which in sequel are matched to elementary hardware resources such as computation storage and networking.

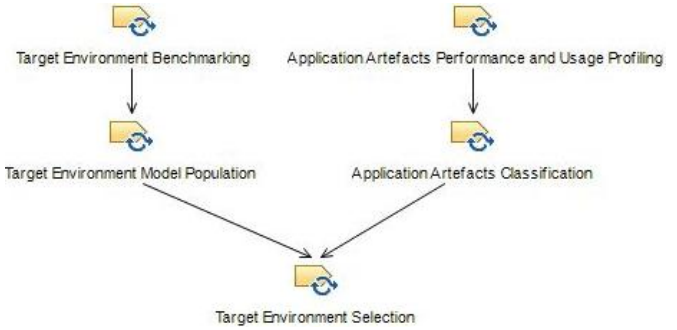

Fig. 5. Target Environment Specifiction EPF Model

In the target environment domain, the various offerings, in the IaaS, PaaS and SaaS layers, are described in a way that facilitates the matching between these offerings and application component requirements. Matching can be performed based on functional aspects (e.g. implementation constraints of PaaS) or on performance ones [13]. Benchmarking will be based on common application types categories (e.g. DB application benchmarks, CPU intensive, I/O intensive etc.). For this purpose a suitable tool is under construction, for semi-automatically launching benchmark tests on target providers in a periodic form to capture also deviation. Metrics are also investigated in order to characterize service offerings based on a combination of performance and cost, in order to improve the overall deployed application's efficiency.

3) Modernization

During the migration phase, once the legacy system has been well understood and decomposed into features and/or components, other tasks such as Model Driven Forward Engineering (MDFE), need to be applied aiming at transforming the legacy system and deploying the migrated one into the target Cloud. During the understanding phase, some high abstract level model views of the legacy system have been produced. These views represent, e.g. the platform independent models (PIM) of the legacy system. Different model views highlight several aspects of the legacy system attending to some concerns. A particular feature can be implemented by different components collaborating altogether. Feature or component views are useful since they enable a feature-driven iterative migration across a number of selected features or components. For instance, persistence could be a feature to iterate on: during this iteration all data sources marked for migration are transformed.

In this modernization phase (Fig. 6), some artefacts produced during early phases are taken as input: a) the PIM models of legacy system produced during the understanding phase, b) the architectural constraints (e.g. migration goals) corresponding to the maturity level the application is aimed to reach and c) the models describing existing target platforms where the application can be deployed.

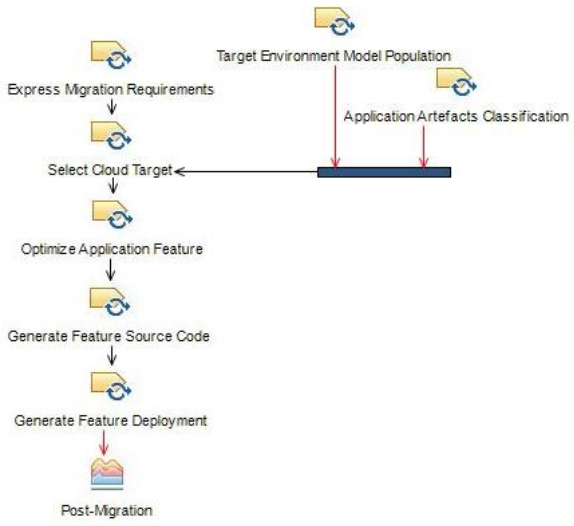

Fig. 6. Modernization EPF Model

Models of the candidate cloud targets environments, which are available through the ARTIST repository are matched

against the afore-posed requirements, using a model-enabled matchmaking algorithm. Models describing those features or components tagged for migration are optimized and transformed, by applying suitable transformation patterns, into target models. These models represent equivalent features or components that are compatible with the selected target provider (e.g. platform deployment/execution compatibility) and that fully exploit the target features and services requested by the application requirements. Following with the example, the persistence feature will be migrated, ensuring behavioural equivalence. Later on, deployment patterns, expressed as model transformations, are used in order to generate specific models from the migrated platform: the deployment descriptors and bundles that configure those required features or services at the cloud target platform.

These migration tasks aim at building and deploying the migrated component corresponding to selected legacy components (e.g. implementations of application features). Once a feature has been migrated, the feature behavioural equivalence (across the legacy and migrated components) and the fulfilment of migration goals (i.e. non-functional requirements) are asserted. If the assessment fails, the process rolls back to the migration phase, in order to re-evaluate the migration requirements and the optimization/transformation strategies. Otherwise, the process moves to the next phase with concerns to the current feature. This migration phase, as stated before, iterates until all features have been migrated.

MDFE strongly relies on models and model transformations e.g. model to model (M2M) and model to text (M2T) for the technical materialization of afore described MDFE tasks. The main aim for these transformations is to generate diverse models providing different views over the migrated application, including its source code. The skeleton of the application is generated automatically, and developers have to write manually some code to complete upgraded and newly added functionalities. Finally, the quality of the migrated system needs to be verified, considering both behavioural (functional) and non-behavioural concerns such as performance or security. The migrated system has to operate similarly to the legacy system and needs to perform at least equally to the old system in terms of functionality and performance. The non-compliance of any of these requirements may cause the non-acceptance of the migrated service by the customers.

In cloud applications the *business model* is tightly intertwined to the technical solution. In the past, companies tended to work first on the technical solution and define after its completion on the pricing, business model and product strategy. Following this approach in cloud-based applications may mean not profiting in its fullness from cloud-based business models and possibly to redefine the whole application architecture to include these business constraints. ARTIST Methodology defines several tasks in order to define the business model, taking as baseline the principles by Osterwalder [17]. Delivering a product is not the same as delivering a service in terms of organizational processes. Existing processes need to be analysed and redefined to adapt the organization to the new structure. Also, new processes related to the service delivery will have to be defined from scratch. The ARTIST methodology has identified several processes that are affected by the shift of business models. These are: *Development Process, Updating Process, SLA Management, Helpdesk, Incidence Management, Marketing, Accountability, Cloud Provider and Roles Alignment.*

E. Post-migration Phase

Once the application has been migrated, several issues need to be considered. A well-defined SaaS application must: *1) Scale, 2) be multitenant, 3) be monitorable, 4) bill automatically and 5) keep the highest security standards*. All these have to be accompanied with a change in the way the company offering the service works: new roles, new service model, a new form of payment, etc., in order to guarantee business continuity. This reorganization, in addition to the model certification to ensure quality of service (described below) will be addressed at this phase (Fig. 7).

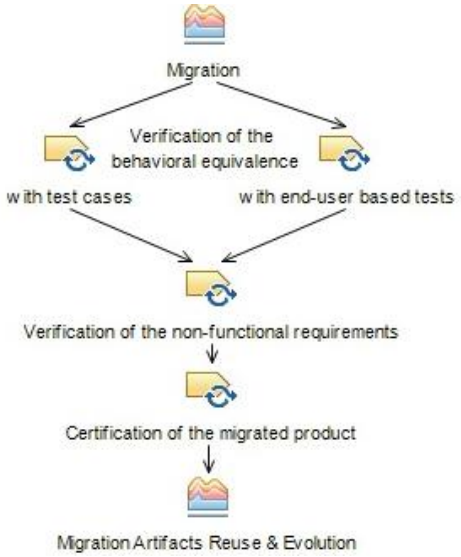

Fig. 7. Post-migration EPF Model

One of the major problems that new service-based software providers have to face is the reluctance of customers to consume new software offered as a service. Providers need to demonstrate their consumers that the service they deliver is of good quality, secure and trustable. Service consumers need to be sure that the SaaS applications they use and consume reach the minimum level of quality that is expected.

To solve this, in the scope of ARTIST, it is proposed to use a Certification Model that analyses:

- The organization (processes, products, financial aspects, and service continuity);
- The service offered (security, administration, support, QoS, SLA, service operational maturity);
- The application (functionality, usability, maintenance).

F. Migration Artefacts Reuse & Evolution Phase

The main purpose of this phase (Fig. 8) is to foster reuse of ARTIST artefacts and to ease the evolution of software to a different cloud provider if needed, to incorporate changes in

the application or even to adapt to a new paradigm that may appear in the future. All intermediate results and artefacts attained in the whole process will be stored in a repository. Its main purpose is to provide a central place to store, archive and organize MDE artefacts such as models, meta-models and transformations and other information produced by the ARTIST tools. Thereby it allows performing expensive operations such as model extraction only once and sharing the results with others.

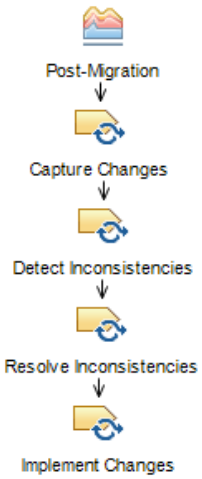

Fig. 8. Migration Artefacts Reuse & Evolution

In the ARTIST process a number of artefacts are produced that are potentially reusable across projects. These include meta-models, generic transformations or format conversions or benchmarking results for various cloud offerings. These artefacts will be made available to the public via a marketplace. The marketplace will allow to publish and consume artefacts as well as to search and browse the available artefacts and will allow users to comment on and rate them.

Evolution of the software during post-migration (or even during migration) will be triggered by new versions of one or more artefacts being submitted to the repository. By supporting traceability in the repository, interested parties can be notified by these changes. They can then decide whether to keep the old version or update to the new version. If they proceed with the new version, the changes are captured after which any inconsistencies in depending artefacts can be detected and in sequel resolved since they might lead to additional changes. Finally, the initial and derived changes need to be implemented in order for the migration workflow to succeed.

V. ARTIST Architecture and Tools

A. *Architecture Overview*

The ARTIST architecture describes an integrated view of the functional blocks or components composing the ARTIST tooling, stereotyped with the main ARTIST methodology block, i.e. pre-migration, migration and post-migration, to introduce the ARTIST components in turn.

The Migration Feasibility Assessment tooling comprises the Maturity Assessment Tool, the Business Feasibility Tool and the Technical Feasibility Tool. These tools interact with each other iteratively to estimate and report about the business and technical migration feasibility. This report is used by the Methodology Process Tool (described in next section) to tailor the particular migration process.

The Migration Feasibility Assessment tooling relies on some of the MDRE features provided by the Model Discovery and Understanding tools, offering low and high level

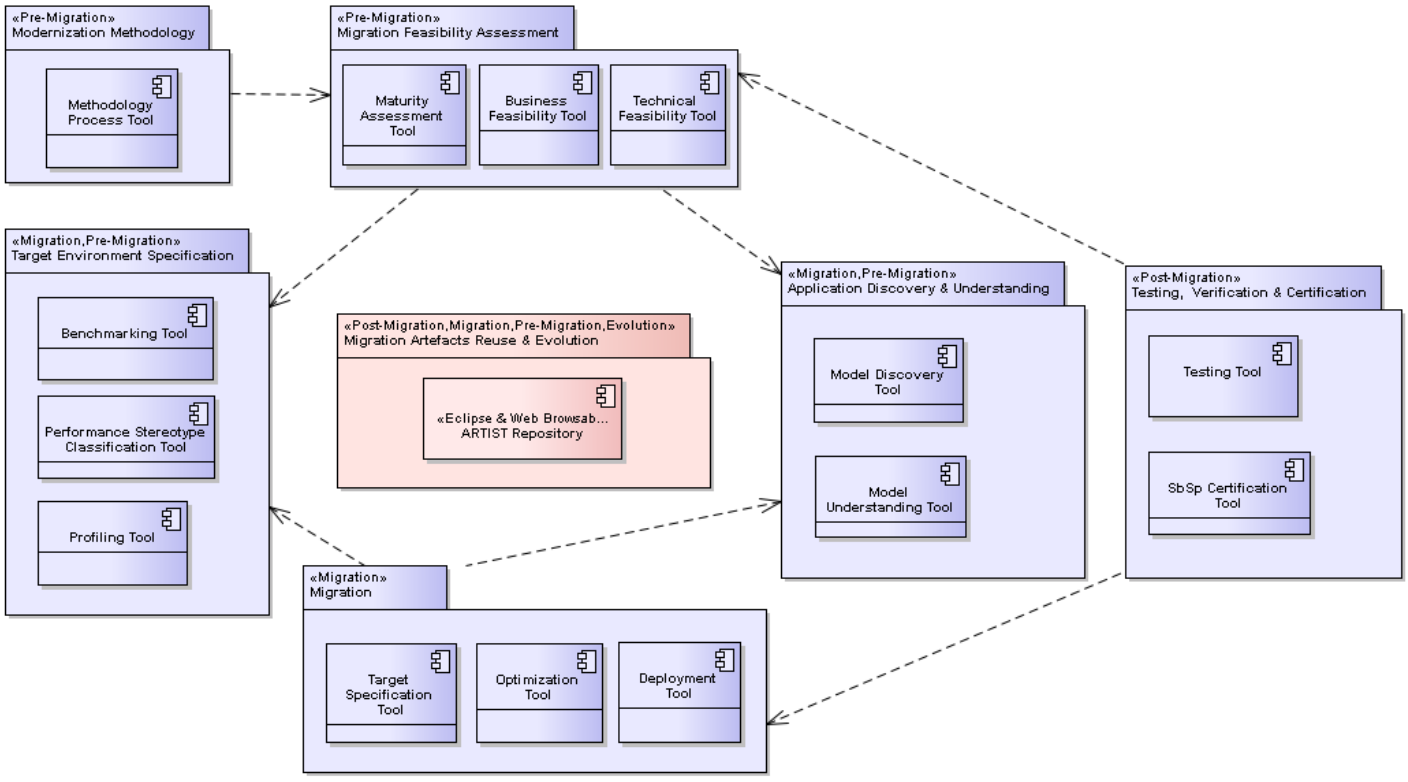

Fig. 9. ARTIST Overall Architecture

abstraction platform-specific and independent models (PSM, PIM) of the legacy application.

The ARTIST Cloud metamodel and their models instances, describing target Cloud providers and offerings, are provided by the Target Environment Specification tooling which comprises the Benchmarking, Performance Stereotype Classification and Profiling tools. These (meta)model artefacts are used during the modernization assessment (e.g. to specify Cloud target and migration requirements) but also during the migration phase by the MDFE tooling (e.g. to specify requirements on the "cloudified" application and the Cloud target environment).

The MDFE Migration tooling comprises Target Specification, Optimization and Deployment tools, which supports the entire migration phase, such as the specification of requirements for the application and the target environment, target lookup and selection, application optimization (e.g. "cloudification") and application building and deployment.

The Testing, Verification and Certification phase is supported by a suite of Testing Tools and the SbSp (Service based Software providers) Certification Tool.

B. ARTIST Methodology Process Tool

The ARTIST Methodology is a modernization and migration solution that covers a wide range of application types, regardless of the underlying technologies, business models, operational modes, etc. In order to practically support this methodology, a core element was incorporated in the overall architecture: the ARTIST Methodology Process Tool. This tool allows the customization and instantiation of the methodology based on the specific application requirements.

The Methodology Process Tool, exploiting the results processed and obtained during the modernisation assessment, defines a customized modernisation process, tailored to the concrete legacy application needs. The tool shows the customized process in detail, its tasks broken down step-by-step, including hooks to invoke the tools required to accomplish each task.

Although the ARTIST tooling mainly relies on the Eclipse Modelling Tools IDE [14], a suitable open source choice due to the well-recognized set of available MDE technologies, the compatibility with Sparx Systems[1] Enterprise Architect, enables ARTIST to also support the MDE-based migration of non-Java legacy applications such as .NET ones for instance.

VI. CONCLUSIONS AND OUTLOOK

In this paper we presented a novel migration approach, which is capable to empower the technical and business capabilities of legacy applications by its modernization and migration to modern cloud environments. The ARTIST Migration Methodology and Framework, as an "all in one" solution allows the legacy applications to exploit the offerings of the cloud providers, enrich their unique characteristics and benefit from offering them as services in a potentially global market. The methodology, covering both the technical and business aspects of the migration process, is tailored to the needs of the specific application and guides the owners and developers in every step so as to take full advantage of their software in an effective and productive manner. In addition, a complete set of tools is supporting each methodology task realizing modern analysis and model-driven engineering approaches, allowing when possible the reuse of artefacts.

ARTIST consortium is currently applying the methodology and the supporting tools to real business scenarios so as to effectively evaluate all technical and business aspects, improve the overall and components specifications and finally to incorporate novel features to the various ARTIST tools that will further enhance their applicability and impact.

[1] *Sparx Systems is member of ARTIST consortium*

ACKNOWLEDGMENT

This work has been supported by the ARTIST Project and has been partly funded by the European Commission under the Seventh (FP7 - 2007-2013) Framework Programme for Research and Technological Development, grant no. 317859.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities", 10th IEEE International Conference on High Performance Computing and Communications, 2008.

[2] P. Mell, T. Grance, "NIST Definition of Cloud Computing", Sp. Publication 800-145, available online at: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[3] Schmidt, Douglas C. "Model-driven engineering." COMPUTER-IEEE COMPUTER SOCIETY- 39.2 (2006): 25.

[4] ARTIST Project: http://www.artist-project.eu/

[5] Menychtas, A. (Ed) (2013). "Deliverable D6.1. Analysis of current migration approaches". Public deliverable.

[6] Grace Lewis, Ed Morris, Liam O'Brien, Dennis Smith, Lutz Wrage, "SMART: The Service-Oriented Migration and Reuse Technique," CMU/SEI-2005-TN-029, September 2005.

[7] MOMOCS Project, Deliverable D3.1b "Methodology Specification", 2008.

[8] Henderson-Sellers, B., & Ralyté, J. (2010). Situational method engineering: state-of-the-art review. Journal of Universal Computer Science, 16(3), 424-478.

[9] Architecture-Driven Modernization (ADM): http://adm.omg.org

[10] REMICS Project, Deliverable D2.4 "REMICS Handbook Interim Release", 2012.

[11] mCloud Project: http://innovacion.grupogesfor.com/web/mcloud

[12] Hugo Brunelière, Jordi Cabot and Grégoire Dupé. How to Deal with your IT Legacy? What is Coming up in MoDisco. In ERCIM News 88 - Special Theme: Evolving Software, pages 43-44, January 2012

[13] George Kousiouris, Dimosthenis Kyriazis, Andreas Menychtas and Theodora Varvarigou, "Legacy Applications on the Cloud: Challenges and enablers focusing on application performance analysis and providers characteristics", 2nd IEEE International Conference on Cloud Computing and Intelligence Systems (IEEE CCIS 2012), Oct. 30th - Nov. 1st, Hangzhou, China

[14] Eclipse Modeling Project http://www.eclipse.org/modeling

[15] Eclipse Process Framework Project (EPF): http://www.eclipse.org/epf/

[16] Software & Systems Process Engineering Metamodel Specification (SPEM) Version 2.0: http://www.omg.org/spec/SPEM/2.0/

[17] Osterwalder, Alexander; Pigneur, Yves; Smith, Allan; et. al. (2009). "Business Model Generation". New Jersey: Hoboken Publication. ISBN 9782839905800.